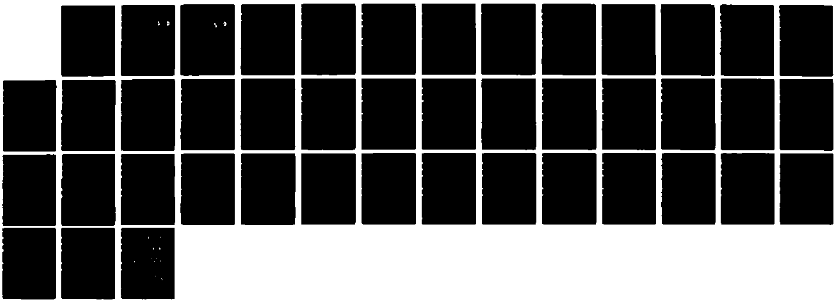
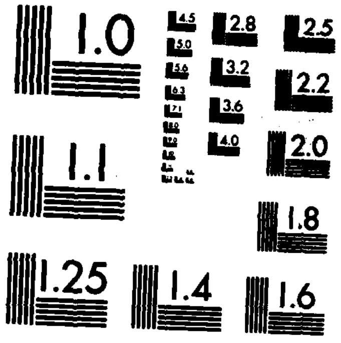
AD-A164 469   COLLISION RESOLUTION ALGORITHMS FOR SPREAD SPECTRUM   1/1
ENVIRONMENTS(U) CONNECTICUT UNIV STORRS DEPT OF
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
UNCLASSIFIED   M PATERAKIS ET AL   JAN 86 UCT/DEECS/TR-86-2 F/G 17/2   NL




1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

(2)

The University of Connecticut

SCHOOL OF ENGINEERING

Storrs, Connecticut 06268

COLLISION RESOLUTION ALGORITHMS FOR SPREAD SPECTRUM ENVIRONMENTS

M. Paterakis and P. Papantoni-Kazakos

January 1986

UCT/DEECS/TR-86-2

Department of

Electrical Engineering and Computer Science

②

COLLISION RESOLUTION ALGORITHMS FOR SPREAD SPECTRUM ENVIRONMENTS

M. Paterakis and P. Papantoni-Kazakos

January 1986

UCT/DEECS/TR-86-2

COLLISION RESOLUTION ALGORITHMS FOR SPREAD SPECTRUM ENVIRONMENTS

by

Michael Paterakis and P. Papantoni-Kazakos
The University of Connecticut
U-157
Storrs, Connecticut 06268

Abstract

In some spread spectrum environments, the low energy of the transmitted signals, in conjuction with the existence of channel noise, do not allow the distinction between collisions and lack of transmissions. For such environments, and for the Poisson user model, the authors ~~we~~ propose and analyze stable full feedback sensing and limited feedback sensing synchronous transmission algorithms. they ~~We~~ assume binary SNS (success versus nonsuccess) feedback per slot, and the possibility of transmission of phony data by a central node. The highest throughput attained by both the full feedback sensing and the limited feedback sensing algorithms is 0.322, while the latter induces somewhat higher delays. This is compensated by the robustness of the limited feedback sensing algorithm in the presence of feedback errors (in contrast to the full feedback sensing algorithm), and its modest requirements on the sensed feedback history. [illegible]

This work was supported by the U.S. Office of Naval Research, under the contract N00014-85-K-0547.

1. Introduction

We consider multi user spread spectrum systems, for data transmission. In such systems, low energy data packets are transmitted to a central node, which has broadcasting capabilities. Let us require that the time of the transmission channel be slotted, where each slot accommodates a single packet. The users of the system act independently; thus, simultaneous transmissions within a single slot may occur, resulting in collisions. In addition, due to the low energy of the data packets, in conjunction with the presence of noise in the transmission channel, make collision slots indistinguishable from empty slots (slots unoccupied by transmissions). That is, the central node can distinguish only between successful slots (slots occupied by a single packet) and nonsuccessful slots. The resulting broadcast feedback per slot is then binary SNS (success versus nonsuccess).

We consider two classes of multi user spread spectrum systems, whose common characteristics are as described in the above paragraph:

(i) Systems consisting of a number of well identified static users.

(ii) Systems consisting of mobile users. In the first class, the users have the capability to maintain in memory the overall broadcast feedback history. In the second class, and due to their mobility, the users are exposed to the broadcast feedback only while they are active; that is,from the time they generate a data packet, to the time that this packet is successfully transmitted. Therefore, full feedback sensing transmission algorithms are applicable in the first class, while only limited sensing such algorithms may be considered for the second class. For both classes, we will initially adopt the Poisson user model (infinitely large number of independent Bernoulli users), since this model represents a limit, and since algorithms designed for it are robust in the presence of changing user structures. For this user model, we will propose and analyze both full feedback sensing and limited feedback sensing transmission algorithms, in the presence of binary SNS feedback.

Given some transmission algorithm, its throughput is defined as the maximum expected traffic that the algorithm maintains, with finite expected delays. For the Poisson user model, the throughput is some intensity, λ^* packets |slot, of the Poisson traffic process. The algorithm is called stable, if its throughput, θ^*, is larger than zero, and the interval $(0,\theta^*)$ is then the stability region of the algorithm. The delay induced by the algorithm is a random variable, and it is defined as the time interval between the instant when a packet is generated, to the instant when it is successfully transmitted. The expected value of this variable is the expected per packet delay induced by the algorithm. In packet networks, the algorithm is called synchronous, if the channel time is divided into slots, each of length equal to a single packet, and packet transmissions can only start at the beginning of some slot.

Let us consider the class of synchronous stable algorithms, for the Poisson user model. Then, among the existing full feedback sensing algorithms within this class, Gallager's algorithm [1] provides the highest throughput, 0.487, and it requires ternary (success versus collision versus empty) feedback per slot. The limited feedback sensing algorithms within the above class were first introduced by Tsybakov and Vvedenskaya [2]. In [3], higher throughput (0.42) such algorithms were proposed and analyzed, for both ternary and CNC (collision versus noncollision) binary feedbacks. In [5], the highest throughput known limited feedback sensing algorithms were proposed and analyzed, for both CNC binary and ternary feedbacks; the throughput for CNC binary feedback is 0.45, and the throughput for ternary feedback is 0.487. In contrast to the algorithm in [1], the algorithms in [6] are robust in the presence of channel errors, and they induce uniformly good delay characteristics. In [4], a unified methodology for the delay analysis of random multiple access algorithms is presented. This methodology has been used for the delay analysis of the algorithms in [3] and [5], as well as for the same analysis of Gallager's algorithm and the controlled ALOHA algorithm. The SNS binary feedback in the full feedback sensing environment was first considered by Berger et al [6].

2. The Model- A Common Algorithmic Rule

The transmissions from the users are assumed synchronous. That is, each user may attempt the transmission of a packet, starting only at the beginning of some slot. Time is measured is slot units, and slot T occupies the time interval [T,T+1). When no packet is transmitted in slot T, the slot is empty. When a single packet is instead transmitted in the interval [T,T+1), it is assumed that it is received correctly by the central node, and slot T is then a success slot. Finally, if at least two packets are simultaneously transmitted in [T,T+1), a collision event occurs, it is then assumed that the information in the involved packets is lost, and retransmission is then necessary.

It is assumed that the central node can distinguish between success versus nonsuccess (emptiness or collision) events per slot, and that it broadcast this SNS binary outcome at the end of each slot. It is assumed that the latter broadcast is received by the users correctly, and without delay. Finally, it is assumed that the central node can transmit phony-noninformation including-packets, whenever necessary. Such transmissions are necessary to help users distinguish between collision versus emptiness events, with some delay. Without such distinction, no stable transmission algorithms exist, for the Poisson user model. This latter model consists of an overall Poisson traffic process, with intensity, λ packets |slot.

In the sequel, we will denote by x_T, the broadcasted by the central node outcome of slot T. We will denote by $x_T= S$, the success event, and we will denote by $x_T=NS$, the nonsuccess event. Also, we will assume that the central node transmits its phony packets synchronously (whenever it does). Whenever such a transmission does not overlap with some transmission from a user, a success event occurs, and the corresponding outcome, S, is then broadcasted. Otherwise, a collision event occurs, and the outcome NS is then broadcasted. Whenever the transmission of a phony packet by the central node is required by the operations of an algorithm, the corresponding algorithmic rule will appear as, TRANSMIT P.

The main problem in the presence of the SNS binary feedback is asynchronicity in channel history information, among the active users. In particular, let $x_t = S; t \leq T-1$, and $x_T = NS$. Then, at time T+1, the active users are divided into two groups, I and II. Group I, if nonempty, contains those users who transmitted in slot T. Group II contains the remaining active users. At T+1, the users in group I know that T is a collision slot; the users in group II, on the other hand, can only deduce that T is either an empty or a collision slot. This asynchronicity is resolved by all the algorithms in this paper, via the followimg common actions in slot T+1.

(i) Each user who transmitted in slot T, transmits again.

(ii) Each user who did not transmit in slot T, withholds transmissions.

(iii) TRANSMIT P.

Due to the above actions, it is clear that $x_{T+1} = NS$, if T is a collision slot, and $x_{T+1} = S$, if T is an empty slot. Therefore, at time T+2, all the active users have the same information about the outcome of slot T, at the expense of one lost slot. In particular,

1. If $x_T = NS$ and $x_{T+1} = NS$, then at T+2 all the active users know that T is a collision slot (termed $x_T = C$).
2. If $x_T = NS$ and $x_{T+1} = S$, then at T+2 all the active users know that T is an empty slot (termed $x_T = E$).

If $T' \geq T+2$ is the first slot after slot T+1, such that $x_{T'} = NS$, then the actions (i), (ii), and (iii) are repeated. Thus, the following, common to all our algorithms, algorithmic rule clearly evolves:

Rule A — If $x_{T-j} = NS$, for all j such that $0 \leq j \leq 2k$, where $k \geq 0$, and $x_{T-2k-1} = S$, then:

(I) In slot T+1:

(a) Each user who transmitted in slot T, transmits again.

(b) Each active user who did not transmit in slot T, withholds transmission.

(c) TRANSMIT P.

(II) At time T+2:

(a) If x_{T+1}=NS, then x_T=C is deduced.

(b) If x_{T+1}=S, then x_T=E is deduced.

(c) Upon deduction of the outcome x_T, the outcome x_{T+1} is eliminated from the memory of each active user. The algorithm operates then on the deduced outcome x_T.

Due to rule A, the algorithms in this paper basically operate with ternary feedback, at the expense of some slots wasted for synchronization among all the active users. Such waste clearly results in throughput reduction and delay increase, as compared to the corresponding algorithms which operate directly with ternary feedback.

3. Full Feedback Sensing Algorithms- The Poisson User Model

The full feedback sensing algorithms require that each user know the channel feedback history at every point in time. Due to that, and in conjuction with rule A, the independent users can then act synchronously, to resolve collisions. In this section, we will consider two algorithms, for the model in section 2. The first algorithm, called FFS1, is Gallager's algorithm, as explained in [7], in conjuction with rule A. The second algorithm, called FFS2, is Capetanakis' dynamic algorithm [8], in conjuction with rule A. Both algorithms use some optimal arrival interval Δ, and in their analysis we will denote by λ the intensity of the Poisson traffic process.

3.1 The FFS1 Algorithm

This algorithm (as explained in [7])utilizes absorptions on the arrival axis. We thus define the parameters L_N and W_N as follows.

W_N: The portion of the initial arrival interval, Δ, that is resolved by the algorithm before absorption, given that the multiplicity of the initial collision in Δ is N.

L_N: The expected number of slots needed by the algorithm to resolve the collision in the portion W_N of the initial arrival interval Δ, before the initial collision is observed.

From Gallager's algorithm in [7], in conjuction with rule A, in section 2, we easily conclude that the following recursions hold.

$$L_0 = 2 \;,\; L_1 = 1$$

$$L_2 = \begin{cases} 4 \;;\; \text{w.p. } 0.5 \\ 2 + L_2 \;;\; \text{w.p. } 0.5 \end{cases}$$

$$L_N = \begin{cases} 2 + L_N & ; \text{ w.p. } 2^{-N} \\ 3 + L_{N-1} & ; \text{ w.p. } N2^{-N} \\ 2 + L_\ell & ; \text{ w.p. } \binom{N}{\ell}2^{-N},\ \ell \neq 0,1 \end{cases} \quad ;N>2 \tag{1}$$

$$W_0 = W_1 = 1$$

$$W_N = \begin{cases} 2^{-1} + 2^{-1}W_{N-i} & ; \text{ w.p. } \binom{N}{i}2^{-N},\ i=0,1 \\ 2^{-1}W_i & ; \text{ w.p. } \binom{N}{i}2^{-N},\ i \neq 0,1 \end{cases} \quad ;N\geq 2 \tag{2}$$

From (1) and (2), we respectively obtain,

$$L_N = \begin{cases} [1-2^{-N+1}]^{-1}[2+N2^{-N}L_{N-1} + \sum_{\ell=1}^{N} \binom{N}{\ell}2^{-N}L_\ell] & ; N>2 \\ 6 \quad ; \quad N=2 \end{cases} \tag{3}$$

$$W_N = 2^{-1}[1-2^{-N}]^{-1}[N2^{-N}W_{N-1} + \sum_{i=0}^{N-1} \binom{N}{i}2^{-N}W_i] \quad ; N\geq 2 \tag{4}$$

Let us define,

$$x = \lambda\Delta$$

$$L(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!} e^{-x} L_k \tag{5}$$

$$W(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!} e^{-x} W_k$$

Then, the stability requirement on the algorithm clearly imposes the condition, $L(x) \leq \lambda^{-1} x W(x)$, and the throughput, λ^*, is then computed as,

$$\lambda^* = \sup_x [xW(x)L^{-1}(x)] \tag{6}$$

If x^* is the value that attains the supremum in (6), then, the optimal initial arrival interval, Δ^*, equals $x^*(\lambda^*)^{-1}$. In appendix A, we compute λ^* and Δ^*. We find,

$$\begin{aligned} \lambda^* &= 0.321946 \\ \Delta^* &= 3.944 \end{aligned} \tag{7}$$

We will perform the delay analysis of the algorithm in section 5, together with the same analysis for the remaining algorithms in this paper.

3.2 <u>The FFS2 Algorithm</u>

As in [8], this algorithm also utilizes an initial optimal interval, Δ, on the arrival axis. Let us define,

L_k: The expected number of slots needed by the algorithm to resolve an initial collision of multiplicity k, <u>before</u> this collision is observed.

Then, the operations of the algorithm in [8], in conjuction with rule A, in section 2, clearly induce the following recursions.

$$L_0 = 2 \ , \ L_1 = 1$$
$$\underset{k \geq 2}{L_k} = \begin{cases} 2 + L_\ell + L_{k-\ell} & ; \text{ w.p. } \binom{k}{\ell} 2^{-k} \ , \ \ell \neq 0 \\ 2 + L_k & ; \text{ w.p. } 2^{-k} \end{cases} \tag{8}$$

From (8), we obtain,

$$L_k = \begin{cases} 2 & ; \ k=0 \\ 1 & ; \ k=1 \\ [1-2^{-k+1}]^{-1} \ [2(1+2^{-k}) + \sum_{\ell=1}^{k-1} \binom{k}{\ell} 2^{-k} \ L_\ell] & ; \ k \geq 2 \end{cases} \tag{9}$$

Let us define,

$$L(x) = e^{-x} \sum_{k=0}^{\infty} \frac{x^k}{k!} L_k \qquad , \quad x = \lambda\Delta \tag{10}$$

Then, the inequality, $L(x) \leq \lambda^{-1} x$, represents the stability condition for the algorithm, and induces the following expressions for the throughput, λ^*, and the optimal initial arrival interval, Δ^*.

$$\lambda^* = \sup_x [x\, L^{-1}(x)] = x^*\, L^{-1}(x^*) \quad , \quad \Delta^* = x^* (\lambda^*)^{-1} \qquad (11)$$

In appendix A, we exhibit the computation of the parameters λ^* and Δ^*. We found,

$$\lambda^* = 0.30062$$
$$\Delta^* = 4.158 \qquad (12)$$

The delay analysis of the algorithm will be performed in section 5.

4. A Limited Feedback Sensing Algorithm - The Poisson User Model

The limited feedback sensing algorithms require that each user follow the channel feedback history only from the time he generates a packet, to the time that this packet is successfully transmitted. That is, each user follows the feedback only while he is active. In this section, we consider an algorithm, named LFS, which is basically an adaptation of the LSTFA algorithm in [5], for the SNS binary feedback. This adaptation is not as straightforward as with the algorithms in section 3. The simple superposition of rule A, in section 2, on the LSTFA, applies only when the latter is in class 2 (see [5]). We thus present here the full description of the LFS algorithm.

4.1 The Description of the LFS Algorithm

The algorithm is performed by each user independently, and at each point in time, it distributes the newly arrived and the nontransmitted packets across two classes, A and B. Transitions in time within and across the classes are controlled by the operations of the algorithm. Class A contains those packets which can not yet decide if the system is empty, while class B contains those packets which know that the system is empty. All packets in class B can thus act synchronously, while the packets in class A can not.

Each nontransmitted packet follows the rules of the algorithm, utilizing a set of parameters, $R, \Delta, L_A, T_1, T_g, \ell$, and G_A. Among those, parameters R and Δ are system parameters, and they are subject to optimization, for the satisfaction of the desired throughput versus expected delay tradeoff. Parameters L_A, T_1, T_g, ℓ, and G_A are recursively updated, as dictated by the algorithmic rules. The above parameters can be interpreted as follows.

Δ : An initial arrival interval.

R : An upper bound to the number of consecutive empty slots allowed during a collision resolution interval, where $R \geq 1$. When a packet observes R + 1 consecutive empty slots, it knows that there is no collision resolution in process. The pattern corresponding to R + 1 empty slots is here represented by R + 1 consecutive slot pairs, each with outcomes (NS,S).

G_A: The number of consecutive empty slots observed by the packet, after it moves to class $\mathcal{B}$. The consecutive empty slot patterns are as in the interpretation of the parameter R.

L_A: The number of slots containing packets from class $\mathcal{A}$, from the time when the packet was generated, to the current time. If the slot within which the packet was generated contains packets from class $\mathcal{A}$, then it is included in the number L_A.

T_g: The time elapsed from the time instant when the packet was generated, to the current time, minus the examined after the above time instant interval.

ℓ : The total length of the arrival interval, which is transmitted in the current slot.

T_1: The time length between the instant when the packet was generated, and the ending point of the most recent arrival interval currently chosen for transmission (see figure 1). All the packets in the interval that corresponds to the length T_1 belong to class $\mathcal{B}$.

We also define the parameter T, as follows.

T : The time elapsed from the time instant when the packet was generated, to the current time.

Upon arrival, each packet initiates the algorithm independently, following the rules below. Simultaneously, step (I) (c) of rule A is followed by the central node.

a. Initialization

Let a packet arrive at the time instant t_a, $t_a \varepsilon [T'-1,T')$. The packet observes then the feedback $x_{T'-1}$, and continuously observes all the feedbacks from this point on, until it is successfully transmitted. At T', the packet moves to class $\mathcal{A}$ below, with initial values $T=T'-t_a$, $L_A=0$, and $G_A=0$.

b. <u>Class $\mathcal{A}$</u>:

Each user has a register with 2(R+1) spaces. Each user with a packet for transmission, who belongs to class $\mathcal{A}$, observes the feedback time sequence and,

1. If the feedback sequence S,S,S appears, then he sets $L_A=2$, and he moves to class $\mathcal{B}$.
2. If the feedback sequence S,S,NS,S appears, then he sets $L_A=3$, and he moves to class $\mathcal{B}$.
3. If the feedback sequence S,NS,NS,S,S appears, then he sets $L_A=4$, and he moves to class $\mathcal{B}$.
4. If R+1 consecutive feedback pairs (NS,S) appear, then he sets $L_A=2R+1$, and moves to class $\mathcal{B}$.

c. <u>Class $\mathcal{B}$</u>:

```
All packets in class $\mathcal{B}$ act as follows:
Start with $T_g=T$, $G_A=0$, and,
2. $\ell = \Delta$, $T_1 = T-L_A$
   Then,
   2.1 If $T_g-(T-T_1)\leq \ell$
   2.1.(a) Set $T_g\to T_g+1$, set $T\to T+1$, set $G_A=0$ and TRANSMIT
    2.1.1 If $X_T=S$, the packet is successfully transmitted.
    2.1.2 If $X_T=NS$, then set $L_A\to L_A+2$,
    2.1.2.(a) Set $\ell\to\ell/2$, and,
     2.1.2.1 If $T_g-(T-T_1)\leq \ell$, move to step 2.1(a)
     2.1.2.2 If $T_g-(T-T_1)>\ell$, set $T_g\to T_g+1$, set $T\to T+1$, and,
      2.1.2.2.1 If $X_T=S$, set $T_g\to T_g-\ell$, set $L_A\to L_A+1$, and move to step 2.1.(a)
       2.1.2.2.2 If $X_T=NS$, set $T_g\to T_g+1$, set $T\to T+1$, set $L_A\to L_A+1$, and,
        2.1.2.2.2.1 If $X_T=NS$, set $L_A\to L_A+1$, and move to step 2.2.1.2.
        2.1.2.2.2.2 If $X_T=S$, set $L_A\to L_A+1$, set $T_g\to T_g-\ell$,
                    set $G_A=G_A+2$, and,
                    If $G_A<2R$, move to step 2.1.2.(a)
                    If $G_A=2R$, move to step 2.1.(a)
   2.2 If $T_g-(T-T_1)>\ell$, set $T_g\to T_g+1$, $T\to T+1$, and,
    2.2.1 If $X_T=NS$, set $T_g\to T_g+1$, $T\to T+2$, $L_A\to L_A+1$.
     2.2.1.1 If $X_T=S$, set $T_g\to T_g-\ell$, $G_A=G_A+2$, and
```

```
update $L_A$ as follows: $L_A \rightarrow L_A+1$. Then,
  If $L_A=4$, then $L_A=3$
  If $L_A=6$, then $L_A=3$
  If $L_A=2R+3$, then $L_A=2R+1$
   and move to step 2.
        2.2.1.2 If $X_T=NS$, set $L_A \rightarrow L_A +1$, set $G_A=0$, and,
        2.2.1.2.(a) Set $\ell \rightarrow \ell/2$
        2.2.1.2.(b) Set $T_g \rightarrow T_g+1$, set $T \rightarrow T+1$, and,
         2.2.1.2.1 If $X_T=S$, set $T_g \rightarrow T_g-\ell$, set $L_A \rightarrow L_A+1$
                     set $T_g \rightarrow T_g+1$, set $T \rightarrow T+1$, set $G_A=0$, and,
          2.2.1.2.1.1 If $X_T=S$, set $T_g \rightarrow T_g-\ell$ , set $L_A=4$, and move to step 2.
          2.2.1.2.1.2 If $X_T=NS$, set $T_g \rightarrow T_g+1$, $T \rightarrow T+1$, $L_A \rightarrow L_A+1$.
          2.2.1.2.1.3 Move to step 2.2.1.2
         2.2.1.2.2 If $X_T=NS$, set $T_g \rightarrow T_g+1$, $T \rightarrow T+1$, $L_A \rightarrow L_A+1$
          2.2.1.2.2.1 If $X_T=NS$, set $L_A \rightarrow L_A+1$, and move to step 2.2.1.2.(a)
          2.2.1.2.2.2 If $X_T=S$, set $T_g \rightarrow T_g-\ell$, set $L_A \rightarrow L_A+1$, set $G_A=G_A+2$ and,
           2.2.1.2.2.2.1 If $G_A<2R$, move to 2.2.1.2.(a)
           2.2.1.2.2.2.2 If $G_A=2R$, move to 2.2.1.2.(b)
      2.2.2 If $X_T=S$, set $T_g \rightarrow T_g-\ell$, and update $L_A$ as follows:
            If $L_A=3$, then $L_A=4$.
            If $L_A=2R+1$, then $L_A=2R+2$.
            Otherwise, set $L_A=2$.
            Set $\ell=\Delta$, $T_1 \rightarrow T-L_A$ and,
       2.2.2.1 If $T_g-(T-T_1)\le\ell$, move to 2.1.(a)
       2.2.2.2 If $T_g-(T-T_1)>\ell$, set $T_g \rightarrow T_g+1$, $T \rightarrow T+1$ and,
        2.2.2.2.1 If $X_T=S$, set $L_A=2$, and move to step 2.
        2.2.2.2.2 If $X_T=NS$, set $L_A \rightarrow L_A+1$, $T_g \rightarrow T_g+1$, $T \rightarrow T+1$.
         2.2.2.2.2.1 If $X_T=NS$, set $L_A \rightarrow L_A+1$ , and move to step 2.2.1.2.
         2.2.2.2.2.2 If $X_T=S$, set $L_A=3$, and move to step 2.
```

In figure 2, we include the flowchart of the LFS algorithm. We point out that the algorithm is a modification of the FFS1 algorithm. The core element in the modification is the initial observation that in the FFS1, a collision resolution interval ends with two consecutive successful transmissions. Based on this observation, a packet operating with the LFS, recognizes that the system is empty, if one of the following feedback sequences occurs: (i) S,S,S (ii) S,S,NS,S (iii) S,NS,NS,S,S. The first sequence corresponds to two consecutive successes. The second sequence corresponds to two consecutive successes, followed by an empty slot. The third sequence provides the information that two consecutive successes occured.

4.2 System Stability

Let us consider the evolution of the LFS algorithm, as seen by an outside observer. Let the time T be measured in slot units, and let the operation of the algorithm start at T=0. At T=2R+2, there will be then 2R+1 slots containing packets from class A, and one slot containing packets from class B. Let us define the variables T_n, D_n, $L_n(A)$, and I_n, as follows.

$T_0=2R+2$

T_n ; $n\geq 1$: The first time after T_{n-1}, such that the distribution of the packets in the unexamined interval is Poisson.

D_n ; $n\geq 0$: The total length of arrival intervals containing packets from class B, at time T_n, where $D_0=1$.

$L_n(A)$; $n\geq 0$: The number of slots containing packets from class A, at time T_n.

$$I_n = \begin{cases} 1 & ; \text{ if } X_{T_{n-1}}=S,\ X_{T_{n-2}}=S \\ 2 & ; \text{ if } X_{T_{n-1}}=NS,\ X_{T_{n-2}}=S,\ X_{T_{n-3}}=S \\ 3 & ; \text{ if } X_{T_{n-1}}=S,\ X_{T_{n-2}}=NS,\ X_{T_{n-3}}=S \\ 4 & ; \text{ if } X_{T_{n-1}}=S,\ X_{T_{n-2}}=NS,\ X_{T_{n-3}}=NS,\ X_{T_{n-4}}=S \end{cases}$$

The triple $S_n \overset{\Delta}{=} (D_n, L_n(A), I_n)$ describes the state of the system at time T_n, as induced by the operation of the algorithm, and the sequence $\{S_n\}$ is a Markov chain. In addition,

$$2 \leq L_n(A) \leq 2R \quad ; \forall n \geq 0$$

$$D_n \geq 0 \quad ; \quad \forall n \geq 0$$

and the values of D_n are denumerable. Given n, given a state value $s_n \overset{\Delta}{=} (d_n, \ell_n(A), I_n)$, then, if $d_n \geq \Delta$, an arrival interval of length Δ is chosen by the algorithm for transmission. If, on the other hand, $d_n < \Delta$, then no packet knows the value d_n, and each packet in class B selects $\ell = \Delta$. Let us define,

$$H_n = T_{n+1} - T_n \tag{13}$$

Then, the set $\{H_n\}$ includes i.i.d. algorithmic sessions, and as in section 5, reference [5], it can be shown that,

$$E\{H_n | d_n < \Delta\} \leq E\{H_n | d_n = \Delta\} = E\{H_n | d_n \geq \Delta\} < \infty \ ; \forall \ \Delta < \infty$$

Consider the time T_n, let the value of the system state be then $s_n = (d_n, \ell_n(A), I_n)$, and let us denote $\Delta_n \overset{\Delta}{=} \min(d_n, \Delta)$. At time T_{n+1}, the algorithm examines then a subset, δ_n, of Δ_n, for transmission. For H_n as in (13), and following the same procedure as in section 5 of reference [5], we conclude that for stability of the Markov chain $\{S_n\}$, it is necessary and sufficient that,

$$E\{\delta_0 | s_0 = (\Delta, \ell(A), I)\} > E\{H_0 | s_0 = (\Delta, \ell(A), I)\} \tag{14}$$

Given the parameters R and Δ, the throughput $\lambda^*(\Delta, R)$ of the LFS algorithm is then the maximum intensity of the Poisson input traffic, that maintains the condition in (14). Optimizing then with respect to Δ, we conclude that given R, the throughput, $\lambda^*(R)$, of the algorithm is as follows, where Δ^*_R denotes the optimal initial arrival interval.

$$\lambda^*(R) = \sup_{\Delta} \lambda^*(\Delta, R) = \lambda^*(\Delta^*_R, R) \tag{15}$$

In appendix A, we exhibit the method for the computation of the throughput $\lambda^*(R)$. In table 1 below, we include the values $\lambda^*(R)$, and the corresponding values Δ^*_R, for various values of the parameter R. As R increases, the throughput increases as well. The increase of the value R may result in increase of the expected per packet delays, however. This last issue will be further discussed in section 5, where the delay analysis of the LFS will be included. The throughput $\lambda^*(6)$ is the same with the throughput $\lambda^*(\infty)$.

R	$\lambda^*(R)$	Δ_R^*
1	0.293376	4.05623
2	0.315648	3.960099
3	0.32043	3.90099
4	0.32158	3.9803028
5	0.321871	3.976739
6	0.321943	3.97585

Table 1

The Throughput of the LFS Algorithm

5. Delay Analysis - The Poisson User Model

In this section, we study the expected per packet delays induced by all the three algorithms in sections 3 and 4. The delay analyses for all the three algorithms are based on the following powerful regeneration theorem. The same theorem was used in [4], for the delay analysis of the Gallager and the controlled ALOHA algorithms, and was also used in [3] and [5], for the delay analysis of the corresponding algorithms.

Theorem A

Let the discrete time process $\{X_n\}_{n\geq 1}$ be regenerative with respect to the renewal process $\{R_i\}_{i\geq 1}$. Let $C_i = R_{i+1}-R_i$, $i\geq 1$, denote the length of the ith regeneration cycle, and let f(.) be some nonnegative real valued measurable function. Then,

If $C \overset{\Delta}{=} E\{C_1\} < \infty$, and $S \overset{\Delta}{=} E\{\sum_{i=1}^{C_1} f(X_i)\} < \infty$, we also have,

$$\lim_{n\to\infty} n^{-1}\sum_{i=1}^{n} f(X_i) = \lim_{n\to\infty} n^{-1}E\{\sum_{i=1}^{n} f(X_i)\} = SC^{-1}, \text{ w.p.1}$$

If in addition to the finiteness of C and S, the distribution of C_1 is also aperiodic, then X_i converges in distribution to X_∞, and,

$$E\{ f(X_\infty)\} = SC^{-1}$$

We point out that the process $\{X_n\}_{n\geq 1}$ is called regenerative with respect to the renewal process $\{R_i\}_{i\geq 1}$, iff for every positive integer M and every sequence $t_1,\dots,t_M$, such that $0 \leq t_1 \leq \dots \leq t_M$, the joint distribution of $X_{t_1+R_i},\dots, X_{t_M+R_i}$ is independent of i.

The random variables $R_i, i \geq 1$, and $C_i = R_{i+1} - R_i$, $i \geq 1$, are then respectively called regeneration points and regeneration cycles. Under the conditions in theorem A, the limiting average and the mean of the limiting distribution of $\{f(X_n)\}_{n \geq 1}$ exist, coincide, and are finite. In addition, their common value is then given in terms of the per cycle quantities, S and C.

Referring now to the three algorithms in this paper, we will define as $\{X_n\}_{n \geq 1}$ in theorem A, the delay process induced by the corresponding algorithm. In each case, we will show that this process is regenerative with respect to a renewal process $\{R_i\}_{i \geq 1}$, with easily identified regeneration points R_i. Then, we will use theorem A, to establish the existence of steady state delays, and to compute the first order steady state moments (expected delays), by appropriately selecting the function $f(\cdot)$. For each one of the three algorithms, we will replace the process $\{X_n\}_{n \geq 1}$ in theorem A, by the process $\{\mathcal{D}_n\}_{n \geq 1}$, where,

$\mathcal{D}_n$: The delay experienced by the nth arrived packet. That is, the time between its arrival and its successful transmission.

$$D_n \triangleq E\{\mathcal{D}_n\} \quad , \; D \triangleq \lim_{n \to \infty} D_n \qquad (16)$$

5.1 The Delays of the FFS1 Algorithm

Consider the operation of the FFS1 algorithm. Let T be a time instant that corresponds to the beginning of slot T, and it is such that all packets that arrived in $(0, t_T)$, where t_T is some time instant such that $t_T < T$, have been successfully transmitted, and there is no information about the arrivals in $[t_T, T)$. Then, T is called a collision resolution instant (CRI), and $d_T = T - t_T$ is called the "lag at T." At every CRI, the algorithm restarts itself by selecting an initial arrival interval $U_T = \min(\Delta, d_T)$. Let us define the sequence $\{T_i\}_{i \geq 1}$ of time points, as with the "0.487" algorithm in [4]. Specifically, let $T_1 = 1$, $d_1 = 2$, and let T_{i+1} be the first instant (corresponding to the beginning of some slot) after T_i, such that $d_{T_{i+1}} = 2$. Let R_i , $i \geq 1$, denote the number of packets successfully transmitted in the interval $(0, T_{i+1}]$. Then, $C_i = R_{i+1} - R_i$, $i \geq 1$, is the number of packets successfully transmitted in the interval $(T_i, T_{i+1}]$, and the sequence $\{R_i\}_{i \geq 1}$ is a renewal process, since $\{C_i\}_{i\geq 1}$ is a sequence of nonnegative i.i.d. random variables. In addition, the delay process, $\{\mathcal{D}_n\}_{n\geq 1}$, of the algorithm is regenerative with respect to the process $\{R_i\}_{i\geq 1}$. Thus, applying theorem A, with function f(•) the identity function, we have that, if,

$$C \overset{\Delta}{=} E\{C_i\} < \infty \quad \text{and} \quad S \overset{\Delta}{=} E\{\sum_{i=1}^{C_1} \mathcal{D}_i\} < \infty \quad \text{, then,}$$

$$D = S\,C^{-1}\text{ , w.p.1.} \qquad (17)$$

;where D is as in (16). It thus suffices to compute the expected values in (17), and prove that they are finite, where we also have,

$$C = \lambda H \quad , \quad H \overset{\Delta}{=} E\{T_{i+1}-T_i\} \qquad (18)$$

We call the expected value H in (18), mean cycle length, and we call the expected value S in (17), mean cumulative delay. In appendix B, we exhibit a method for the computation of upper and lower bounds on the mean cycle length and the mean cumulative delay. Those bounds are then used for the derivation of upper and lower bounds-respectively denoted D^u and D^ℓ - on the mean packet delay, D, in (16). The latter bounds are exhibited in table 2 below, for various Poisson intensities, within the stability region of the algorithm. We point out that using the same method, tighter bounds can be devised, with additional computational effort.

λ	D^ℓ	D^u
0.01	2.06868	2.082206
0.05	2.44738	2.52379
0.09	2.815948	3.120604
0.13	3.244687	4.02865
0.21	5.28434	7.34515
0.25	10.03156	14.98144
0.29	22.57134	48.518118
0.32	289.41968	599.96109

Table 2

Mean Packet Delays for the FFS1 Algorithm

5.2 The Delays of the FFS2 Algorithm

For this algorithm, the CRIs, and the sequences $\{T_i\}_{i\geq 1}$, $\{R_i\}_{i\geq 1}$, and $\{C_i\}_{i\geq 1}$, are defined exactly as with the FFS1 algorithm in section 5.1, and so are then the quantities, C, S, D, and H, in (17) and (18), and the function f(•). Again, theorem A applies, if the mean cycle length, H, and the mean cumulative delay, S, are both bounded. Then, the mean packet delay, D, equals $\lambda^{-1}H^{-1}S$, where λ is the intensity of the Poisson traffic process. In appendix B, we derive upper and lower bounds on the quantities H and S. Through them, we derive the upper and lower bounds, D^u and D^ℓ, on the mean packet delay D, for various values of the Poisson intensity λ, within the stability region of the algorithm. We include those bounds in table 3 below. Again, tighter bounds can be found, via the same method, and with additional computational effort.

λ	D^ℓ	D^u
0.01	2.0885448	2.0885665
0.05	2.5160811	2.5241058
0.09	3.0976268	3.1320109
0.13	3.935235	4.0316111
0.21	7.7381153	8.296022
0.25	14.049199	15.641169
0.29	67.385942	79.315479

Table 3

Mean Packet Delays for the FFS2 Algorithm

5.3 The Delays of the LFS Algorithm

The sequences, $\{T_i\}_{i\geq 1}$, $\{R_i\}_{i\geq 1}$, and $\{C_i\}_{i\geq 1}$, the function f(.), and the quantities D, H, and S, are again defined as in section 5.1. The delay process $\{\mathcal{D}_n\}_{n\geq 1}$ is again regenerative with respect to the renewal process $\{R_i\}_{i\geq 1}$, and theorem A applies. The mean cycle length, H, and the mean cumulative delay, S, are of course computed differently here, and if bounded, the mean packet delay, D, equals $\lambda^{-1}H^{-1}S$. In appendix B, we exhibit the method for computing upper and lower bounds, $D^u(R)$ and $D^\ell(R)$, on the mean packet delay D, for every value of the algorithmic parameter R. In table 4 below, we include computed values of the bounds $D^u(1)$ and $D^\ell(1)$ - that correspond to algorithmic parameter R equal to one - for various values of the intensity, λ, of the Poisson input traffic. The computation of the bounds $D^u(R)$ and $D^\ell(R)$, for $R \geq 2$ requires increasing computational effort.

λ	$D^\ell(1)$	$D^u(1)$
0.01	3.05538	3.2257
0.05	3.38136	4.4241
0.10	3.9288	6.6706
0.15	4.81452	10.5137
0.20	6.6218	18.4668
0.25	12.7931	45.4634
0.27	23.8604	94.0532
0.28	46.3069	192.844

Table 4

Mean Packet Delays for the LFS Algorithm with R=1.

As the value of the algorithmic parameter R increases, the mean packet delays for small Poisson intensities, λ, increase as well. Given R_1 and R_2 such that $R_1 > R_2$, there exists some Poisson intensity, λ^o, however, such that for Poisson intensities above λ^o, the mean packet delays induced when the logarithmic parameter is R_2, exceed those induced by the parameter value R_1. This is shown in figure 3, where the upper bounds $D^u(1)$ and $D^u(2)$ - respectively corresponding to R=1 and R=2 - are plotted against λ. The crossing point is then approximately equal to 0.2. That is, for Poisson rates in (0,0.2), the LFS with R=1 is superior, while for such rates in (0.2, 0.315), the LFS with R=2 is recommended.

5.4 Comparative Discussion

In figure 4, we plot the lower bound, D^ℓ, on the mean packet delay, against the Poisson traffic intensity λ, for the FFS1, FFS2, and LFS with R=1 algorithms. The upper bounds in tables 3,4, and 5 exhibit similar behavior. The FFS1 algorithm induces the lowest mean packet delays, for every λ value within its stability region. It is sensitive to channel errors (leading to deadlocks), however, and it requires full feedback sensing. The LFS algorithm, on the other hand, is robust in the presence of channel errors (as the algorithm in [5]), and only requires limited feedback sensing, at the expense of somewhat increased mean packet delays. Since full feedback sensing is impossible in mobile spread spectrum environments, the latter algorithm is then indispensible.

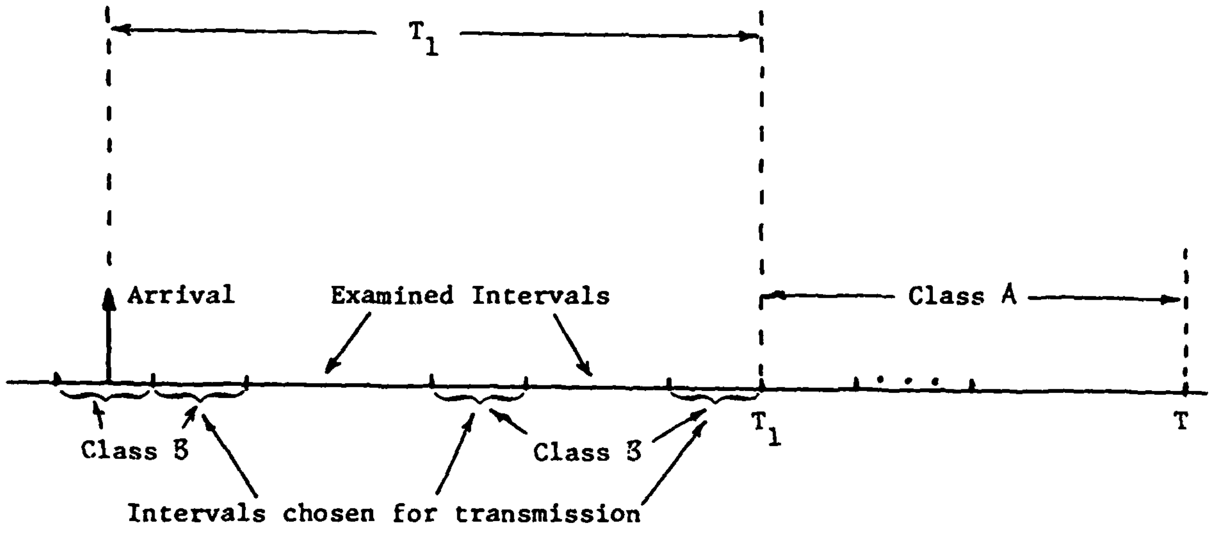

T_1
Arrival
Examined Intervals
Class A
Class B
Class B
T_1
T
Intervals chosen for transmission

Figure 1

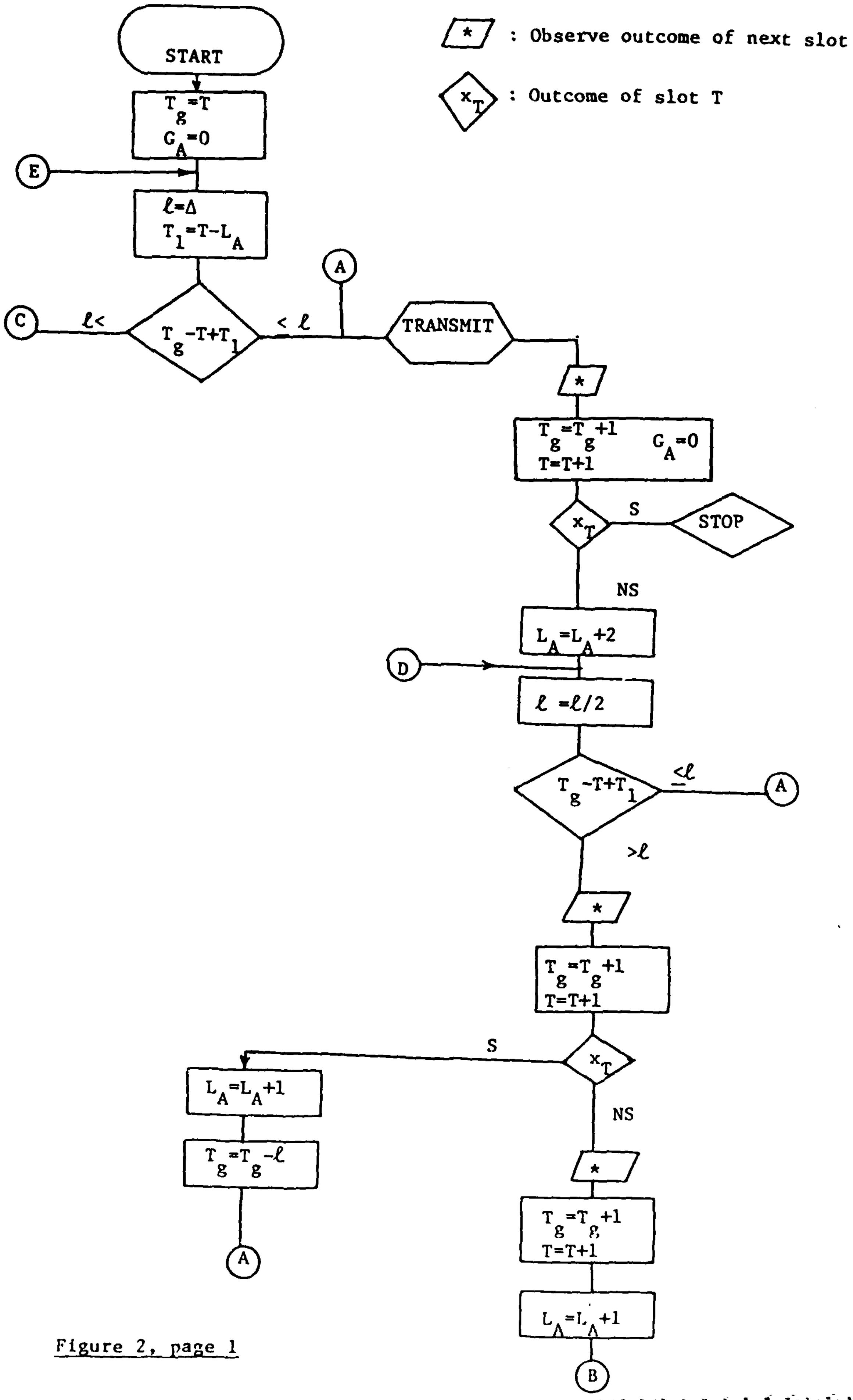


Figure 2, page 1

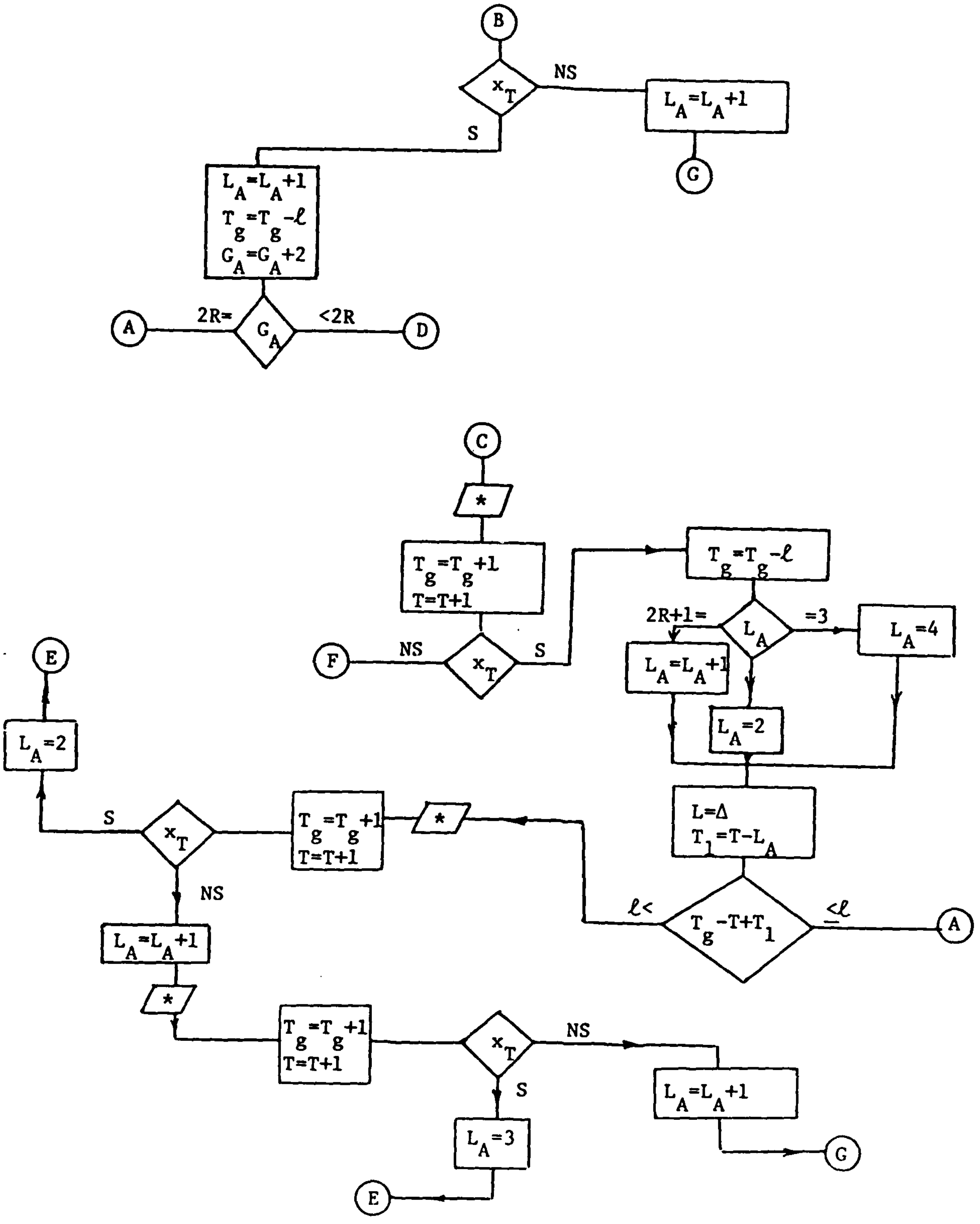


Figure 2, page 2

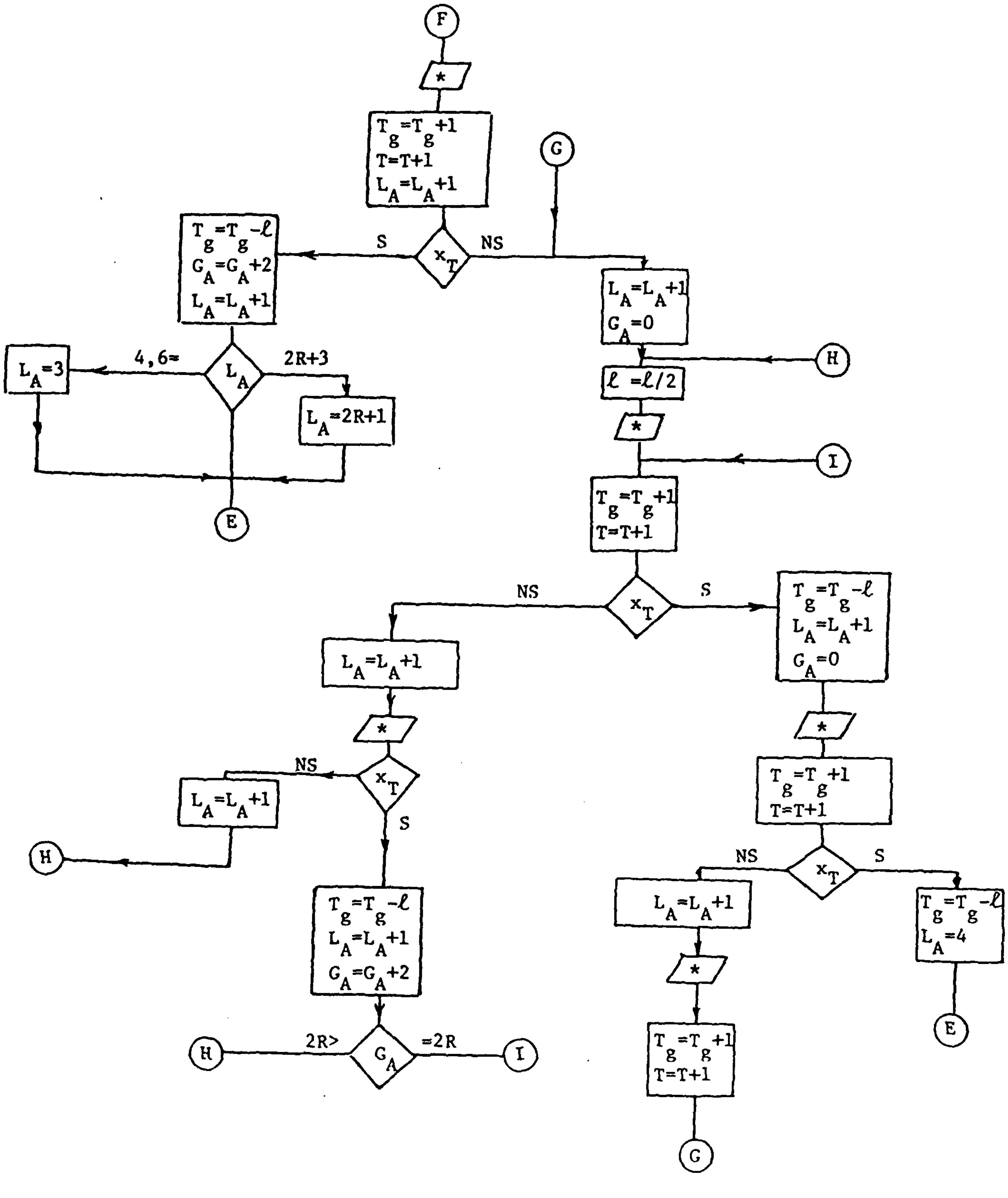


Figure 2, page 3

The Flowchart of the LFS Algorithm

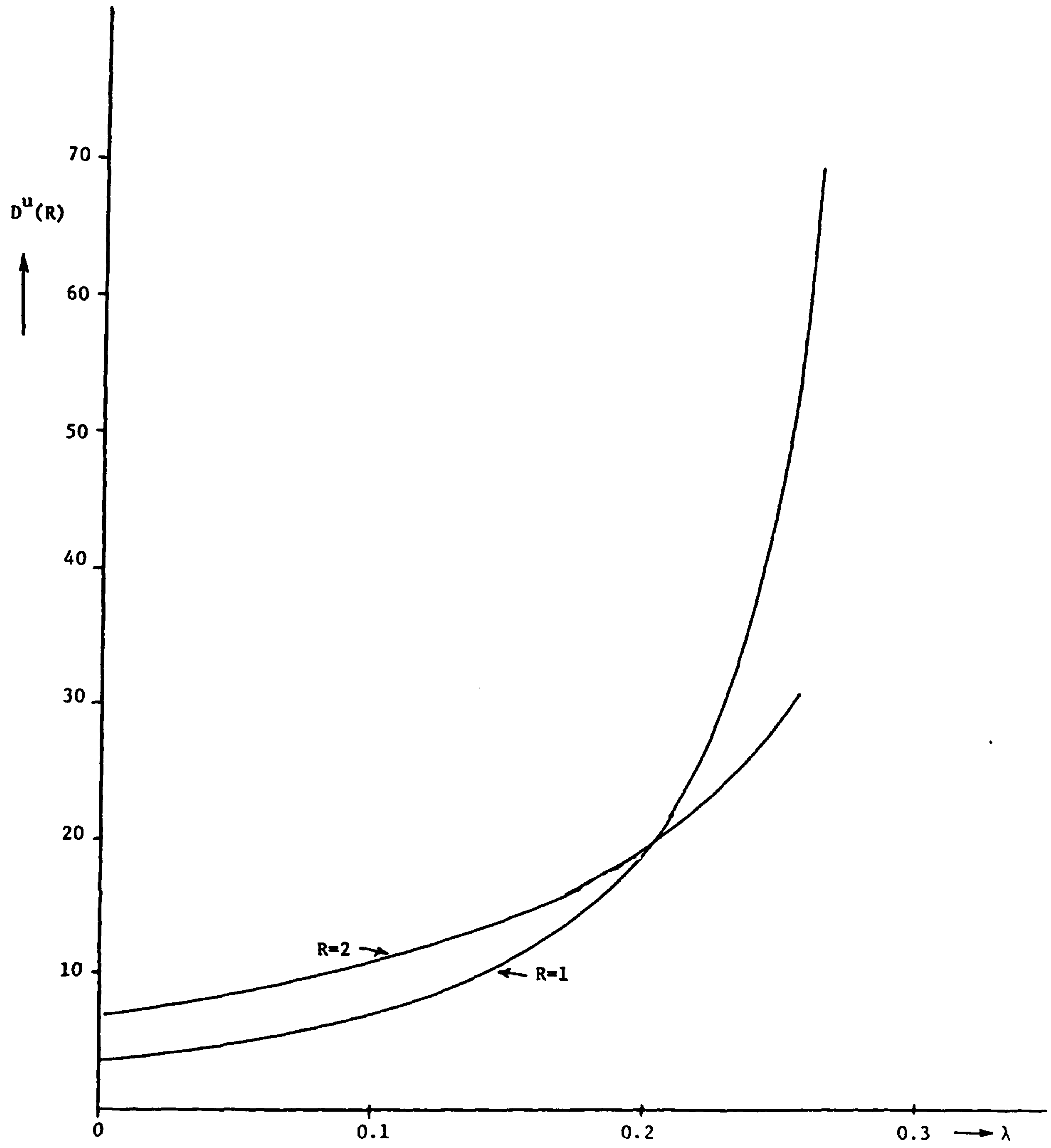


Figure 3

Mean Packet Delays for the LFS Algorithm

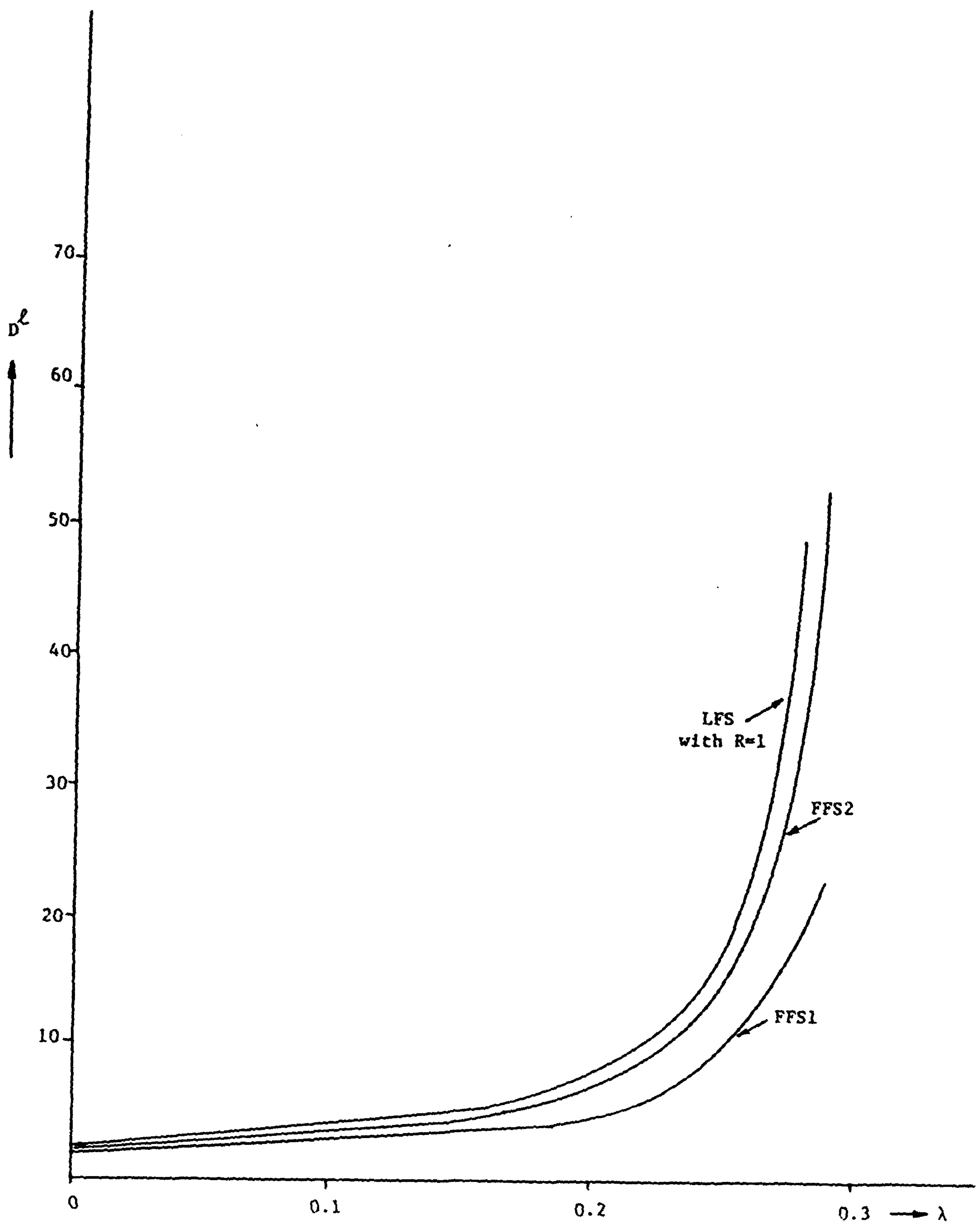


Figure 4

Comparative Delays

Appendix A

A.1 The Throughput of the FFS1 Algorithm

We use the following lemma, for the set $\{L_k\}$in (1) and (3).

Lemma A1

The inequality $L_k \leq 6k$ holds, for every $k \geq 1$.

Proof

We prove the lemma by induction. We clearly have $L_k < 6k$, for k=1, 2. Accepting then that $L_\ell \leq 6\ell$; $1 \leq \ell \leq k$, and using (3), we easily find that $L_{k+1} \leq 6(k+1)$. The proof is now complete.

Let now k_o be some positive integer, and let us then adopt the following simple bounds.

$$\begin{aligned} 1 \leq L_k \leq 6k \quad &; \quad \forall k \geq k_o \\ 0 \leq W_k \leq 1 \quad &; \quad \forall k \geq k_o \end{aligned} \tag{A.1}$$

Using (A.1) and (5), we then obtain,

$$\begin{aligned} e^x B^\ell(x) \triangleq x \sum_{k=0}^{k_o} \frac{x^k}{k!} W_k \leq x \sum_{k=0}^{\infty} \frac{x^k}{k!} W_k = e^x\, x\, W(x) \leq \\ \leq x \left(\sum_{k=0}^{k_o} \frac{x^k}{k!} W_k + \sum_{k=k_o+1}^{\infty} \frac{x^k}{k!} \right) = \\ = x \left(e^x + \sum_{k=0}^{k_o} \frac{x^k}{k!} (W_k - 1) \right) \triangleq e^x B^u(x) \end{aligned} \tag{A.2}$$

$$\begin{aligned} e^x L^\ell(x) \triangleq \sum_{k=0}^{k_o} \frac{x^k}{k!} L_k \leq \sum_{k=0}^{\infty} \frac{x^k}{k!} L_k = e^x L(x) \leq \sum_{k=0}^{k_o} \frac{x^k}{k!} L_k + \sum_{k=k_o+1}^{\infty} 6k \frac{x^k}{k!} = \\ = 6xe^x + L_o + \sum_{k=1}^{k_o} \frac{x^k}{k!} (L_k - 6k) \triangleq e^x L^u(x) \end{aligned} \tag{A.3}$$

Then,

$$\frac{B^\ell(x)}{L^u(x)} \leq \frac{xW(x)}{L(x)} \leq \frac{B^u(x)}{L^\ell(x)} \tag{A.4}$$

We selected $k_o=20$, we computed precisely from (3) the values L_k ; $k \leq 20$, and we then found numerically the suprema of the ratios, $B^{\ell}(x)/L^u(x)$ and $B^u(x)/L^{\ell}(x)$. Those suprema were both attained at $x^*=1.266$, and were identical to each other to the fifth decimal point, and equal to λ^* in (7).

A.2 The Throughput of the FFS2 Algorithm

It is easily proven that lemma A1 holds. Then, for some positive integer k_o, and using (10), we conclude that expression (A.3) holds here as well, for the set $\{L_k\}$ in (9). We selected $k_o=20$, and we computed the values L_k ; $k \leq 20$ precisely from (9). Then, we found numerically the suprema of the expressions, $x/L^u(x)$ and $x/L^{\ell}(x)$, where $x/L^u(x) \leq x/L(x) \leq x/L^{\ell}(x)$. The latter suprema are both attained at $x^*=1.25$, and they are identical to the fifth decimal point, and equal to λ^* in (12).

A.3 The Throughput of the LFS Algorithm

Let us define,

$$\Omega \triangleq \Delta^{-1}\delta_0$$

$$L_k \triangleq E\{H_0 \mid D_0 = \Delta, \text{ k packets in } \Delta\} \qquad \text{(A.5)}$$

$$W_k \triangleq E\{\Omega \mid D_0 = \Delta, \text{ k packets in } \Delta\}$$

$$p_i^k \triangleq \binom{k}{i} 2^{-k}$$

Then, from the operation of the algorithm, we deduce the following recursions.

$$L_0 = 2,\ L_1 = 1$$

$$\underset{;k\geq 2}{L_k} = \begin{cases} 2+2\ell + 1 + L_{k-1} & ;\ \text{w.p. } (p_0^k)^{\ell} p_1^k\ ,\ 0 \leq \ell \leq R-1 \\ 2+2\ell + L_i & ;\ \text{w.p. } (p_0^k)^{\ell} p_i^k\ ,\ 0 \leq \ell \leq R-1,\ k \geq i \geq 2 \\ 2+2R + L_k & ;\ \text{w.p. } (p_0^k)^R \end{cases} \qquad \text{(A.6)}$$

$$W_0 = W_1 = 1$$

$$\underset{;k\geq 2}{W_k} = \begin{cases} \sum_{j=1}^{\ell} 2^{-j} + \Delta 2^{-\ell-1} + 2^{-\ell-1} W_{k-1} & ;\ \text{w.p. } (p_0^k)^{\ell} p_1^k,\ 0 \leq \ell \leq R-1 \\ \sum_{j=1}^{\ell} 2^{-j} + 2^{-\ell-1} W_i & ;\ \text{w.p. } (p_0^k)^{\ell} p_i^k\ ,\ 0 \leq \ell \leq R-1, k \geq i \geq 2 \\ \sum_{j=1}^{R} 2^{-j} + 2^{-R}\, W_k & ;\ \text{w.p. } (p_0^k)^R \end{cases} \qquad \text{(A.7)}$$

From (A.6) and (A.7), we respectively find, where $\sum_{i=\ell}^{k} x_i \overset{\Delta}{=} 0$, if $k<\ell$,

$$L_k = \begin{cases} 2 & ; k=0 \\ 1 & ; k=1 \\ \left\{1-(p_0^k)^R - p_0^k \dfrac{1-(p_0^k)^R}{1-p_0^k}\right\}^{-1} \left\{ 2 + \dfrac{1-(p_0^k)^R}{1-p_0^k} \left[2\, p_0^k + p_1^k L_{k-1} + \sum_{i=1}^{k-1} p_i^k L_i \right] \right\} \end{cases} \quad \text{(A.8)}$$

$$W_k = \begin{cases} 1 & ; k=0,1 \\ \left\{1-2^{-R}(p_0^k)^R - [2-p_0^k]^{-1}[1-2^{-R}(p_0^k)^R]p_0^k\right\}^{-1}[2-p_0^k]^{-1}[1-2^{-R}(p_0^k)^R]\,. & \\ \qquad \cdot \left\{p_0^k + p_1^k + p_1^k W_{k-1} + \sum_{i=2}^{k-1} p_i^k W_i\right\} ; & k\geq 2 \end{cases} \quad \text{(A.9)}$$

Let us define,

$$L(x) \overset{\Delta}{=} \sum_{k=0}^{\infty} e^{-x} \frac{x^k}{k!} L_k$$

$$W(x) \overset{\Delta}{=} \sum_{k=0}^{\infty} e^{-x} \frac{x^k}{k!} L_k$$

; where $0 \leq W_k \leq 1$; $\forall$ k, and where lemma A.1 is found to hold on the sequence $\{L_k\}$. Then, the inequalities in (A.2) and (A.3) hold here as well, and so does (A.4), where again,

$$\lambda^* = \sup_x \left(\frac{x\, W(x)}{L(x)}\right)$$

For all the R values in table 1, and for $k_o = 20$ in (A.2) and (A.3), the suprema of the bounds in (A.4) where indentical to each other to the fifth decimal, and equal to the corresponding values $\lambda^*(R)$ in table 1. The values Δ_R^* were found as, $\Delta_R^* = x_R^* (\lambda^*(R))^{-1}$.

Appendix B

B.1 Delays for the FFS1 Algorithm

Consider the CRIs, T, the instants t_T, and the intervals U_T, in section 5.1. Then, in slot T, some arrival interval $[t_{T'}, t_T+U_T)$ is transmitted, where after some random number, ℓ, of slots, another CRI, T' is reached, with a corresponding, $t_{T'}>t_T$. Let us define the following quantities.

ℓ_d :	The number of slots needed to resolve an arrival interval between two subsequent CRIs, given that the initial length of this interval is d.
w_d :	The length of the actually "examined" interval before absorption, given that the length of the initial interval is d.
$P(\ell,w\|d)$:	The probability that, given an initial length d interval between two subsequent CRIs, the actually "examined" before absorption interval has length w, and ℓ slots are needed for its resolution.
H_d :	The expected number of slots needed to reach lag equal to 2, starting from lag equal to d.

The operations of the algorithm induce then the following recursions.

$$H_d = \begin{cases} 2 & ;\ \text{w.p. } P(2,d|d) \\ E\{\ell_d\} + H_{d-w+\ell} & ;\ \text{w.p. } P(\ell,w|d) \end{cases}, \text{ for } d \leq \Delta \tag{B.1}$$

$$H_d = E\{\ell_\Delta\} + H_{d-w+\ell} \quad ;\ \text{w.p. } P(\ell,w|\Delta), \text{ for } d > \Delta$$

Taking expectations in (B.1), we obtain,

$$H_d = \begin{cases} E\{\ell_d\} + \sum\limits_{\substack{\ell,w \\ \ell\neq 2}} H_{d-w+\ell}\, P(\ell,w|d) & ;\ d \leq \Delta \\ E\{\ell_\Delta\} + \sum\limits_{\ell,w} H_{d-w+\ell}\, P(\ell,w|\Delta) & ;\ d > \Delta \end{cases} \tag{B.2}$$

The expected value, H, in (18) is equal to H_2, where H_d is given by (B.2). We thus need to find bounds on H_2. Following the same procedure as with the "0.487" algorithm in [4], on the linear system in (B.2), we find the following bounds.

$$\begin{aligned} H^{\ell} \triangleq\ & E\{\ell_2\} + A[2-2P(2,2|2)] -P(1,2|2) - E\{w_2-\ell_2\}] + c^{\ell}[1-P(2,2|2) -P(1,2|2)] \\ & + P(1,2|2)[B + c^{\ell}\{1-P(2,1|1)\}] \leq H_2 = H \leq E\{\ell_2\} + \\ & + A[2-2P(2,2|2) -P(1,2|2) -E\{w_2-\ell_2\}] + c^{u}[1-P(2,2|2) -P(1,2|2)] + \\ & + P(1,2|2)[B + c^{u}\{1-P(2,1|1)\}] \triangleq H^{u} \end{aligned} \tag{B.3}$$

where,

$$A \overset{\Delta}{=} E\{\ell_\Delta\} \, [E\{w_\Delta\} - E\{\ell_\Delta\}]^{-1}$$

$$p(d) \overset{\Delta}{=} P^{-1}(2,d|d) \left\{ E\{\ell_d\} + A[E\{\ell_d\} - E\{w_d\}] - 2AP(2,d|d) \right\}$$

$$c^\ell \overset{\Delta}{=} \inf_{1 \le d \le \Delta} p(d) \qquad (B.4)$$

$$c^u \overset{\Delta}{=} \max\,(-A, \sup_{1 \le d \le \Delta} p(d))$$

$$B \overset{\Delta}{=} E\{\ell_1\} + A[E\{\ell_1\} - E\{w_1\} + 1 - 2P(2,1|1)]$$

From (B.3) and (B.4), we computed the upper and lower bounds, H^u and H^ℓ, on H, shown in table B.1 below.

λ	H^ℓ	H^u
0.01	2.01435	2.01731
0.05	2.12193	2.14923
0.09	2.30954	2.364348
0.13	2.514186	2.671167
0.21	3.109886	3.98608
0.25	4.852168	6.60396
0.29	8.09345	10.923344
0.32	111.98070	161.02000

Table B.1

Bounds on H for the FFS1 Algorithm

Let us define the following quantities, referring to the beginning of this appendix.

N : The number of packets in $[t_T, t_{T'})$

ψ : The sum of the delays of the above N packets, after the CRI, T.

z : The sum of the delays of the above N packets, until the instant $t_T + U_T$.

$Z_k \overset{\Delta}{=} E\{z|N=k, U_T=1\} = d^{-1} E\{z|N=k, U_T=d\} \quad ; 1 \le d \le \Delta$

$\psi_k \overset{\Delta}{=} E\{\psi|N=k, U_T=1\} = E\{\psi|N=k, U_T=d\} \quad ; 1 \le d \le \Delta$

S_d : The expected sum of delays of all packets transmitted, when starting from lag d, the algorithm reaches lag 2.

The algorithm induces then the following recursions.

$$S_d = \begin{cases} 0 & ; \ \ell_d = 2 \\ E\{\psi|d\} + E\{z|d\} + S_{d-w+\ell} & ; \text{ w.p. } P(\ell,w|\Delta) \ , \ \ell_d \neq 2 \end{cases} \quad \text{,for } d \leq \Delta \tag{B.5}$$

$$S_d = E\{\psi|d\} + E\{z|d\} + (d-\Delta)N + S_{d-w+\ell} \quad ; \text{ w.p. } P(\ell,w|\Delta) \ , \text{ for } d > \Delta$$

From (B.5), we obtain,

$$S_d = \begin{cases} E\{\psi|d\} + E\{z|d\} + \sum\limits_{\substack{\ell,w \\ \ell\neq 2}} S_{d-w+\ell} P(\ell,w|d) \ ; \ d \leq \Delta \\ E\{\psi|\Delta\} + E\{z|\Delta\} + (d-\Delta)E\{N|\Delta\} + \sum\limits_{\ell,w} S_{d-w+\ell} P(\ell,w|\Delta) \ ; \ d > \Delta \end{cases} \tag{B.6}$$

; where for S as in (17), we have, $S = S_2$. Following the same procedure as with the bounds H^u and H^ℓ on H, we find,

$$S^\ell \leq S_2 = S \leq S^u \tag{B.7}$$

; where,

$$S^u = G + c^u\,[1-P(2,2|2)] + P(1,2|2)\,[s^u - M_1 - M_2 - c^\ell]$$
$$S^\ell = G + c^\ell[1-P(2,2|2)] + P(1,2|2)[s^\ell - M_1 - M_2 - c^u]$$
$$G = E\{\psi|2\} + E\{z|2\} + 2(M_1+2M_2)[1-P(2,2|2)] + M_2E\{(w_2-\ell_2)^2\} - (4M_2+M_1)E\{w_2-\ell_2\}$$
$$s^a = E\{\psi|1\} + E\{z|1\} + M_2\Big(1 + E\{w_1^2\} + E\{\ell_1^2\} - 2E\{\ell_1 w_1\} - 2[E\{w_1\} - E\{\ell_1\}]$$
$$- 4P(2,1|1)\Big) + M_1\Big(1 - E\{w_1\} + E\{\ell_1\} - 2P(2,1|1)\Big) + c^a[1-P(2,1|1)] \ ; \ a=u,\ell$$
$$M_2 = E\{N|\Delta\}\ 2^{-1}\ [E\{w_\Delta\} - E\{\ell_\Delta\}]^{-1}$$
$$M_1 = [E\{w_\Delta\} - E\{\ell_\Delta\}]^{-1}\Big(E\{\psi|\Delta\} + E\{z|\Delta\} + M_2[E\{w_\Delta^2\} + E\{\ell_\Delta^2\} - 2E\{w_\Delta \ell_\Delta\}$$
$$- \Delta E\{N|\Delta\}\Big) \tag{B.8}$$
$$p(d) = P^{-1}(2,d|d)\Big(E\{\psi|d\} + M_2E\{(w_d-\ell_d)^2\} - (2M_2d+M_1)\ E\{w_d-\ell_d\}$$
$$- (4M_2+2M_1)\ P(2,d|d)\Big)$$
$$c^u = \sup_{1\leq d\leq \Delta} p(d)$$
$$c^\ell = \inf_{1\leq d\leq \Delta} p(d)$$

From (B.8), we computed the bounds S^ℓ and S^u, that we include, for various values of λ, in table B.2 below.

λ	S^{ℓ}	S^{u}
0.01	0.04173	0.04194
0.05	0.26299	0.26776
0.09	0.59920	0.64864
0.13	1.12672	1.31674
0.21	4.42339	4.79694
0.25	16.5620	18.17311
0.29	71.5080	113.8769
0.32	14912.7540	21498.9000

Table B.2

Bounds on S for the FFS1 Algorithm

From (17) and (18), we conclude that the bounds D^u and D^{ℓ}, on the mean packet delay, D, are computed from tables B.1 and B.2, as follows:

$$D^u = \lambda^{-1} S^u (H^{\ell})^{-1} \quad , \quad D^{\ell} = \lambda^{-1} S^{\ell} (H^u)^{-1} \qquad (B.9)$$

B.2 Delays for the FFS2 Algorithm

Let us define the following quantities.

ℓ_d : The number of slots needed to resolve an initial arrival interval of length d, where $1 \leq d \leq \Delta$.

$P(\ell|d)$: The probability that $\ell_d = \ell$.

H_d : The expected number of slots needed to reach lag 2, starting from lag d, subject to successful transmission of every packet in the arrival interval of length $\min(d,\Delta)$.

S_d : The expected sum of delays of all packets transmitted, when starting from lag d, the algorithm reaches lag 2, and all the packets in the arrival interval of length $\min(d,\Delta)$ are successfully transmitted.

Let the quantities N, ψ, and z be as in section B.1. Then, the algorithm induces the following recursions.

$$H_d = \begin{cases} 2 & ; \text{w.p. } P(2|d) \\ E\{\ell_d\} + H_{\ell} & ; \text{w.p. } P(\ell|d) \ , \ \ell \neq 2 \end{cases} \quad , \text{ for } d \leq \Delta \qquad (B.10)$$

$$H_d = E\{\ell_d\} + H_{d-\Delta+\ell} \quad ; \text{w.p. } P(\ell|\Delta) \ , \text{ for } d > \Delta$$

$$S_d = \begin{cases} E\{\psi|N=0\} + E\{z|N=0\} & ;\text{ w.p. } P(2|d) \\ E\{\psi|d\} + E\{z|d\} + S_\ell & ;\text{ w.p. } P(\ell|d)\ ;\ell\neq 2 \end{cases} \quad ,\text{ for } d \leq \Delta \tag{B.11}$$

$$S_d = E\{\psi|\Delta\} + E\{z|\Delta\} + (d-\Delta)\, E\{N|\Delta\} + S_{d-\Delta+\ell} \quad ;\text{ w.p. } P(\ell|\Delta)\ ,\text{ for } d > \Delta$$

From (B.10) and (B.11), we respectively obtain,

$$H_d = \begin{cases} E\{\ell_d\} + \sum\limits_{\ell\neq 2} H_\ell\, P(\ell|d) & ;\ d \leq \Delta \\ E\{\ell_\Delta\} + \sum\limits_{\ell} H_{d-\Delta+\ell}\, P(\ell|\Delta) & ;\ d > \Delta \end{cases} \tag{B.12}$$

$$S_d = \begin{cases} E\{\psi|d\} + E\{z|d\} + \sum\limits_{\ell\neq 2} S_\ell\, P(\ell|d) & ;\ d \leq \Delta \\ E\{\psi|\Delta\} + E\{z|\Delta\} + (d-\Delta)\, E\{N|\Delta\} + \sum\limits_{\ell} S_{d-\Delta+\ell}\, P(\ell|\Delta) & ;\ d > \Delta \end{cases} \tag{B.13}$$

Considering the quantities H and S, in section 5.2, we have, $H = H_2$ and $S = S_2$. We are thus seeking upper and lower bounds on the quantities H_2 and S_2, given respectively by (B.12) and (B.13). We note that here,

$$P(\ell|d) = \sum_k P(\ell|k)\, P(k|d)$$

; where,

$$P(k|d) = e^{-\lambda d} \frac{(\lambda d)^k}{k!}$$

$$P(\ell|k) = 2^{-k}\, [P(\ell-2|k) + P(\ell-4|k)] + \sum_{i=1}^{k-1} \binom{k}{i} 2^{-k} \sum_{\substack{\ell_1+\ell_2=\ell-2 \\ \ell_1,\ell_2 \geq 1}} P(\ell_1|i)P(\ell_2|k-i) \tag{B.14}$$

To compute bounds on H_2 and S_2, we use iterations. Let $H_d^{(i)}$ and $S_d^{(i)}$ denote the corresponding ith iterations. Then, we easily conclude,

$$H_d^{(i+1)} = \begin{cases} H_d^{(i)} + \sum\limits_{\ell<K\ ,\ell\neq 2} [H_\ell^{(i)} - H_\ell^{(0)}]\, P(\ell|d) & ;\ d \leq \Delta \\ H_d^{(i)} + \sum\limits_{\ell=1}^{K} [H_{d-\Delta+\ell}^{(i)} - H_{d-\Delta+\ell}^{(0)}]\, P(\ell|\Delta) & ;\ d > \Delta \end{cases} \tag{B.15}$$

; with,

$$Md + c^\ell \leq H_d^{(0)} \leq Md + c^u \tag{B.16}$$

; where,

$$M = [\Delta - E\{\ell_\Delta\}]^{-1}\, E\{\ell_\Delta\}$$

$$c^u = \sup_{1\le d\le\Delta} f(d) \quad , \quad c^\ell = \inf_{1\le d\le\Delta} f(d) \qquad \text{(B.17)}$$

$$f(d) = P^{-1}(2|d)\left(E\{\ell_d\} + M[E\{\ell_d\}-\Delta] - 2MP(2|d)\right)$$

Using K=40, 100 iterations, and via (B.15), (B.16), and (B.17), we found the lower and upper bounds, H^ℓ and H^u, on H_2 (and thus on H), shown in table B.3 below.

λ	H^ℓ	H^u
0.01	2.02178	2.02178
0.05	2.14534	2.14534
0.09	2.33792	2.33793
0.13	2.63508	2.63520
0.21	4.04580	4.05625
0.25	6.40808	6.47551
0.29	26.37999	27.20669

Table B.3

Bounds on H for the FFS2 Algorithm

For the quantities S_d, we started with,

$$e^\ell + M_1 d + M_2 d^2 \le S_d^{(0)} \le M_2 d^2 + M_1 d + e^u \qquad \text{(B.18)}$$

; where,

$$M_2 = 2^{-1}\,[\Delta - E\{\ell_\Delta\}]^{-1}\, E\{N|\Delta\}$$

$$M_1 = [\Delta - E\{\ell_\Delta\}]^{-1}\left(E\{\psi|\Delta\} + E\{z|\Delta\} + M_2[\Delta^2 + E\{\ell_\Delta^2\} - 2\Delta E\{\ell_\Delta\}] - \Delta E\{N|\Delta\}\right)$$

$$e^\ell = \inf_{1\le d\le\Delta} f_o(d) \quad , \quad e^u = \sup_{1\le d\le\Delta} f_o(d) \qquad \text{(B.19)}$$

$$f_o(d) = P^{-1}(2|d)\Big(E\{\psi|d\} + E\{z|d\} + M_2[E\{\ell_d^2\} - 4P(2|d)] + + M_1[E\{\ell_d\} - d - 2P(2|d)]\Big)$$

; and where,

$$E\{N|d\} = \lambda d \;\; ; \; 1\le d\le\Delta \quad , \quad E\{z|d\} = 2^{-1}\lambda d^2 \;\; ; \; 1\le d\le\Delta$$

$$E\{\psi|d\} = \sum_k E\{\psi|k\}P(k|d)$$

$$E\{\psi|k\} = \frac{1}{1-2^{-k+1}}\left(2k + \sum_{i=1}^{k-1}\binom{k}{i}2^{-k}[2E\{\psi|i\} + (k-i)E\{\ell_d|i\}]\right)$$

$$E\{\psi|0\} = 0 \quad , \quad E\{\psi|1\} = 1$$

$$E\{\ell_d^2\} = \sum_k E\{\ell_d^2|k\}\ P(k|d)$$

$$E\{\ell_d^2|k\} = [1-2^{-k+1}]^{-1}\left(4 + 12.2^{-k} + 12.2^{-k}\ E\{\ell_d|k\} + 2\sum_{m=1}^{k-1}\binom{k}{m}2^{-k}[E^2\{\ell_d|m\} + \right.$$

$$\left. + 4E\{\ell_d|m\} + E\{\ell_d|m\}\ E\{\ell_d|k-m\}]\right) \qquad (B.20)$$

The iterations for S_d evolve as those in (B.15). After 100 iterations, we found the upper and lower bounds, S^u and S^ℓ, on S_2 (and thus on S), that are included in table B.4 below.

λ	S^ℓ	S^u
0.01	0.04222	0.04222
0.05	0.26989	0.27075
0.09	0.65178	0.65901
0.13	1.34811	1.38107
0.21	6.59143	7.04845
0.25	22.74395	25.05749
0.29	531.67114	606.77905

Table B.4

Bounds on S for the FFS2 Algorithm

The bounds D^u and D^ℓ are computed from tables B.3 and B.4, and from the expressions in (B.9).

B.3 Delays for the LFS Algorithm

Here, we will derive bounds on the mean packet delay, D, when the algorithmic parameter R equals one. The methodology for $R \geq 2$ is similar. Let the quantities H_d and S_d be defined as in section B.1. Let δ_0 and d_0 be as in section 4.2, and let ℓ_d be defined as in section B.1. Let us also define, where T is some CRI,

N : The number of packets in δ_0.

N_0: The number of packets in $\Delta - \delta_0$.

$P(\ell,w|d)$: The probability that $\ell_d = \ell$, and that given $d=\min(d_0,\Delta)$, then $\delta_0=w$.

ψ : The sum of the delays of the N packets, after T.

z : The sum of the delays of the N packets, before T-1.

Then, the algorithm induces the following recursions.

$$H_d = \begin{cases} E\{\ell_d\} + \sum\limits_{\substack{\ell \neq 2 \\ w}} P(\ell,w|d)\, H_{d-w+\ell} & ; 1 \leq d \leq \Delta \\ E\{\ell_\Delta\} + \sum\limits_{\ell,w} P(\ell,w|\Delta)\, H_{d-w+\ell} & ; d > \Delta \end{cases} \quad \text{(B.21)}$$

$$S_d = \begin{cases} E\{z|d\} + E\{\psi|d\} + E\{N|d\} + E\{N_0\delta_0|d\} + \sum\limits_{\substack{\ell \neq 2 \\ w}} P(\ell,w|d) S_{d-w+\ell} \; ; 1 \leq d \leq \Delta \\ E\{z|\Delta\} + E\{\psi|\Delta\} + E\{N|\Delta\} + \lambda(d-\Delta)\, E\{\delta_0|\Delta\} + E\{N_0\delta_0|\Delta\} + \\ \qquad + \sum\limits_{\ell,w} P(\ell,w|\Delta) S_{d-w+\ell} \; ; d > \Delta \end{cases} \quad \text{(B.22)}$$

The expressions (B.21) and (B.22) both determine infinite dimensionality linear systems. Using the methodology for the delay analysis of the algorithm in [5], we conclude that the systems in (B.21) and (B.22) have both unique solutions, within the class of quadratically bounded sequences, if $E\{\delta_0|d_0=\Delta\} > E\{\ell_\Delta\}$. The latter inequality determines the stability region of the algorithm. In addition, we have again that $H=H_2$ and $S=S_2$, where H and S are the quantities in section 5.3. We derive upper and lower bounds on S_2 and H_2 (and thus on S and H), as follows.

$$H^\ell \leq H_2 = H \leq H^u$$
$$S^\ell \leq S_2 = H \leq S^u \quad \text{(B.23)}$$

; where,

$$H^u = E\{\ell_2\} + M_0\{E\{\ell_2\} - E\{\delta_0|2\} + 2[1-P(2,2|2)]\} + c^u[1-P(2,2|2)]$$

$$H^\ell = H^u - [c^u - c^\ell][1-P(2,2|2)]$$

$$M_0 = [E\{\delta_0|\Delta\} - E\{\ell_\Delta\}]^{-1} E\{\ell_\Delta\}$$

$$c^u = \max\left(-M_0\, ,\, \sup_{1\leq d\leq\Delta} h(d)\right) \quad , \quad c^\ell = \inf_{1\leq d\leq\Delta} h(d)$$

$$h(d) = P^{-1}(2,d|d)\{E\{\ell_d\} + M_0[E\{\ell_d\} - E\{\delta_0|d\}] - 2M_0P(2,d|d)\}$$

$$S^u = E\{z|2\} + E\{\psi|2\} + E\{N|2\} + E\{N_0\delta_0|2\} + M_2[4-4e^{-2\lambda} + E\{(\delta_0-\ell_2)^2|2\} -$$
$$- 4\, E\{\delta_0-\ell_2|2\}] + M_1[2-2e^{-2\lambda} - E\{\delta_0-\ell_2|2\}] + e^u(1-e^{-2\lambda})$$

$$S^{\ell} = S^{u} - [e^{u}-e^{\ell}]\,(1-e^{-2\lambda})$$

$$M_2 = [E\{\delta_0|\Delta\} - E\{\ell_\Delta\}]^{-1}\lambda\, E\{\delta_0|\Delta\}$$

$$M_1 = [E\{\delta_0|\Delta\} - E\{\ell_\Delta\}]^{-1}\Big\{E\{z|\Delta\} + E\{\psi|\Delta\} + E\{N|\Delta\} + E\{N_0\delta_0|\Delta\} - \\ - \lambda\,\Delta\, E\{\delta_0|\Delta\} + M_2\, E\{(\delta_0-\ell_\Delta)^2|\Delta\}\Big\}$$

$$e^{u} = \sup_{1\le d\le\Delta} f(d) \quad , \quad e^{\ell} = \inf_{1\le d\le\Delta} f(d)$$

$$f(d) = P^{-1}(2,d|d)\Big\{E\{z|d\} + E\{\psi|d\} + E\{N|d\} + E\{N_0\delta_0|d\} + \\ + M_2[E\{(\delta_0-\ell_d)^2|d\} - 2d\, E\{\delta_0-\ell_d|d\} - 4P(2,d|d)] \\ - M_1[E\{\delta_0-\ell_d|d\} + 2\,P(2,d|d)]\Big\} \qquad (B.24)$$

As compared to the computations in sections B.1 and B.2, the difficulty here is the computation on the expectations included in the expressions in (B.24). Let us define,

$E_A\{X|d,k\}$: The conditional expectation of the random variable X, given that Δ equals A, given d, where $d \le \Delta$, and given that there are k packets in d.

Then,

$$E_A\{X|d\} = \sum_{k=0}^{\infty} E_A\{X|d,k\}\; e^{-\lambda d}\,\frac{(\lambda d)^k}{k!} \qquad (B.25)$$

Let us define the subset V of (0,1], as follows,

$V = \{v : v=1, \text{ or } v = 2^{-j_1} + 2^{-(j_1+j_2)} + \ldots + 2^{-(j_1+\ldots+j_M)}\ ;\ M, j_1, \ldots, j_M$ positive integers$\}$. Then, the expectations $E_A\{X|d,k\}$, for $k \ge 0$, can be as closely approximated as desired, by selecting $d|A$ values in V, in conjuction with the corresponding recursive expressions. We do not include the latter expressions in this paper, due to lack of space. The interested reader may seek reference [9]. Using the above methodology, for M=6 in the subset V, we computed the values of the bounds H^u, H^{ℓ}, S^u, S^{ℓ} in (B.23), which are included in table B.5 below. The bounds in table 4 were computed from the values in the latter table, in conjuction with the expressions in (B.9).

λ	H^{ℓ}	H^{u}	S^{ℓ}	S^{u}
0.01	1.9984	2.0577	0.06287	0.06446
0.05	1.9244	2.2128	0.374114	0.425699
0.10	1.96223	2.5658	1.00807	1.30892
0.15	2.17837	3.22266	2.32733	3.4354
0.20	2.78712	4.6554	6.16544	10.2939
0.25	5.1692	9.7539	31.1957	58.7521
0.27	9.55336	19.0408	122.667	242.601
0.28	18.4793	37.9514	492.076	997.815

Table B.5

Bounds on S and H for the LFS Algorithm

References

[1] R.G. Gallager, "Conflict Resolution in Random Access Broadcast Networks," in Proc. AFOSR Workshop Commun. Th. Appl., Provincetown, MA., Sept. 1978, pp.17-20.

[2] B.S. Tsybakov and N.D. Vvedenskaya, "Random Multiple-Access Stack Algorithm," Probl. Peredachi Inf., Vol. 16, No. 3, 1980, pp.80-94.

[3] L. Georgiadis and P. Papantoni-Kazakos,"Limited Feedback Sensing Algorithms for the Packet Broadcast Channel," IEEE Trans. Inf. Th.,Special Issue on Random Access Communications, Vol. IT-31, No.2, March 1985, pp. 280-294.

[4] L. Georgiadis, L. Merakos, and P. Papantoni-Kazakos, "A Unified Method for Delay Analysis of Random Multiple Access Algorithms," Univ. Connecticut, EECS Dept., Technical Report UCT/DEECS/TR-85-8, Aug. 1985. Also, submitted for publication.

[5] L. Georgiadis, and P. Papantoni-Kazakos, "A 0.487 Throughput Limited Sensing Algorithm," IEEE Trans. Inf. Th., to appear.

[6] T. Berger, N. Mehravari, D. Towsley, and J. Wolf,"Random Multiple-Access Communication and Group Testing," IEEE Trans. Commun., Vol. COM -32, July 1984, pp. 769-779.

[7] R.G. Gallager, "A Perspective on Multiaccess Channelss," IEEE Trans. Inf. Th., Special Issue on Random Access Communications, Vol. IT-31, No.2, March 1985, pp. 124-142.

[8] J.I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," IEEE Trans. Inf. Th., Vol. IT-25, No.5, Sept. 1979, pp. 505-515.

[9] M. Paterakis, "Transmission Algorithms for Some Multi-User Spread Spectrum Systems," M.S. Thesis, Univ. Connecticut, 1986.